

CFM03348

US

Appln. No. 10/729,007
GAI: NYA.

日本国特許庁
JAPAN PATENT OFFICE

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office.

出願年月日 2003年 7月24日
Date of Application:

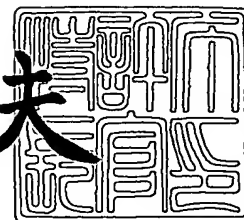
出願番号 特願2003-201162
Application Number:
[ST. 10/C]: [JP 2003-201162]

出願人 キヤノン株式会社
Applicant(s):

2004年 1月 6日

特許庁長官
Commissioner,
Japan Patent Office

今井康夫



出証番号 出証特2003-3108771

【書類名】 特許願

【整理番号】 255622

【提出日】 平成15年 7月24日

【あて先】 特許庁長官殿

【国際特許分類】 G06F 3/00

【発明の名称】 符号データ作成方法及び装置

【請求項の数】 13

【発明者】

 【住所又は居所】 東京都大田区下丸子3丁目30番2号 キヤノン株式会社
社内

 【氏名】 榎田 幸

【特許出願人】

 【識別番号】 000001007

 【氏名又は名称】 キヤノン株式会社

【代理人】

 【識別番号】 100076428

 【弁理士】

 【氏名又は名称】 大塚 康德

 【電話番号】 03-5276-3241

【選任した代理人】

 【識別番号】 100112508

 【弁理士】

 【氏名又は名称】 高柳 司郎

 【電話番号】 03-5276-3241

【選任した代理人】

 【識別番号】 100115071

 【弁理士】

 【氏名又は名称】 大塚 康弘

 【電話番号】 03-5276-3241

【選任した代理人】

【識別番号】 100116894

【弁理士】

【氏名又は名称】 木村 秀二

【電話番号】 03-5276-3241

【先の出願に基づく優先権主張】

【出願番号】 特願2002-356738

【出願日】 平成14年12月 9日

【手数料の表示】

【予納台帳番号】 003458

【納付金額】 21,000円

【提出物件の目録】

【物件名】 明細書 1

【物件名】 図面 1

【物件名】 要約書 1

【包括委任状番号】 0102485

【プルーフの要否】 要

【書類名】 明細書

【発明の名称】 符号データ作成方法及び装置

【特許請求の範囲】

【請求項 1】 サーバが管理する符号データのうち、断片的な第 1 の符号データを格納した格納手段を備えるクライアントにおいて、J P E G 2 0 0 0 符号データを作成する符号データ作成方法であって、

前記クライアントにおいて前記 J P E G 2 0 0 0 符号データの作成に必要なとなる符号データと、前記格納手段に格納された前記第 1 の符号データとから、不足する第 2 の符号データを算出する算出工程と、

前記サーバに対して、算出された前記第 2 の符号データを要求する要求工程と

、

前記サーバから、前記第 2 の符号データを取得する取得工程と、

取得された前記第 2 の符号データを前記格納手段に格納する格納工程と、

取得された前記第 2 の符号データのヘッダ情報を解析して、前記符号データを複数の独立した符号データに分割する分割工程と、

前記分割工程で分割された単位毎に、前記独立した符号データを構成する全ての符号データが前記格納手段に格納されているか否かを判定する判定工程と、

前記独立した符号データを構成する全ての符号データが格納されていない場合、格納されていない部分にダミー符号データを格納するダミー格納工程と、

前記格納手段に格納された符号データを J P E G 2 0 0 0 符号データとして出力する出力工程と

を有することを特徴とする符号データ作成方法。

【請求項 2】 前記判定工程によって、前記独立した符号データを構成する全ての符号データが前記格納手段に格納されていると判定された場合、該格納手段に格納された該符号データを前記分割工程で分割された前記独立した符号データに置き換える置換工程をさらに有することを特徴とする請求項 1 に記載の符号データ作成方法。

【請求項 3】 前記符号データが、パケット単位で取り扱われることを特徴とする請求項 1 又は 2 に記載の符号データ作成方法。

【請求項 4】 前記ダミー符号データが、J P E G 2 0 0 0 で規定されている zero length packet データであることを特徴とする請求項 1 から 3 までのいずれか 1 項に記載の符号データ作成方法。

【請求項 5】 前記分割工程が、所定サイズのタイル単位で前記符号データを分割することを特徴とする請求項 1 から 4 までのいずれか 1 項に記載の符号データ作成方法。

【請求項 6】 前記独立した符号データのそれぞれのヘッダ情報を前記分割工程で分割された前記タイル単位のサイズに変更する変更工程をさらに有することを特徴とする請求項 1 から 5 までのいずれか 1 項に記載の符号データ作成方法。

【請求項 7】 前記クライアントが画像データを表示する表示手段をさらに備え、前記第 1 の符号データが前記画像データの符号データであって、

前記表示手段に表示された画像データの表示領域の移動又は拡大表示に応じて、前記算出工程における前記必要となる符号データを設定する設定工程と、

前記出力工程で出力された前記 J P E G 2 0 0 0 符号データをデコードするデコード工程と、

デコードされた画像データを前記表示手段に画面表示する表示工程と

をさらに有することを特徴とする請求項 1 から 6 までのいずれか 1 項に記載の符号データ作成方法。

【請求項 8】 前記判定工程によって、前記独立した符号データを構成する全ての符号データが前記格納手段に格納されていると判定された場合、前記分割工程で分割された単位ごとに、該単位に関するメインヘッダをそのまま出力し、該単位のインデックスを所定のインデックスに置換し、該単位に含まれる符号化データ本体を前記分割工程で分割された前記独立した符号データに置き換える置換手段をさらに備えることを特徴とする請求項 1 に記載の符号データ作成方法。

【請求項 9】 前記分割工程で分割された少なくとも 2 以上の単位について、前記判定工程、前記ダミー格納工程、前記置換工程及び前記出力工程を並列処理することを特徴とする請求項 2 に記載の符号データ作成方法。

【請求項 10】 第 1 のコンピュータが管理する符号データのうち、断片的

な第1の符号データを格納した格納手段を備える第2のコンピュータにおける符号データ作成装置であって、

前記第1のコンピュータに管理された前記符号データのうち、断片的な第1の符号データを格納する第1の格納手段と、

前記第2のコンピュータにおいて J P E G 2 0 0 0 符号データの作成に必要な符号データと、前記格納手段に格納された前記第1の符号データとから、不足する第2の符号データを算出する算出手段と、

前記第1のコンピュータに対して、算出された前記第2の符号データを要求する要求手段と、

前記第1のコンピュータから、前記第2の符号データを取得する取得手段と、

取得された前記第2の符号データを格納する第2の格納手段と、

取得された前記第2の符号データのヘッダ情報を解析して、前記符号データを複数の独立した符号データに分割する分割手段と、

前記分割手段で分割された単位毎に、前記独立した符号データを構成する全ての符号データが前記第1又は第2の格納手段に格納されているか否かを判定する判定手段と、

前記独立した符号データを構成する全ての符号データが格納されていない場合、格納されていない部分にダミー符号データを格納する第3の格納手段と、

前記第1、第2及び第3の格納手段に格納された符号データを用いて J P E G 2 0 0 0 符号データを作成する作成手段と

を備えることを特徴とする符号データ作成装置。

【請求項11】 前記第1及び第2のコンピュータが、ネットワークを介して互いに通信可能であることを特徴とする請求項9記載の符号データ作成装置。

【請求項12】 第1のコンピュータが管理する符号データのうち、断片的な第1の符号データを格納した格納手段を備える第2のコンピュータに、 J P E G 2 0 0 0 符号データを作成させるためのプログラムであって、

前記第2のコンピュータにおいて前記 J P E G 2 0 0 0 符号データの作成に必要な符号データと、前記格納手段に格納された前記第1の符号データとから、不足する第2の符号データを算出する算出手順と、

前記第1のコンピュータに対して、算出された前記第2の符号データを要求する要求手順と、

前記第1のコンピュータから、前記第2の符号データを取得する取得手順と、

取得された前記第2の符号データを前記格納手段に格納する格納手順と、

取得された前記第2の符号データのヘッダ情報を解析して、前記符号データを複数の独立した符号データに分割する分割手順と、

前記分割手順で分割された単位毎に、前記独立した符号データを構成する全ての符号データが前記格納手段に格納されているか否かを判定する判定手順と、

前記独立した符号データを構成する全ての符号データが格納されていない場合、格納されていない部分にダミー符号データを格納するダミー格納手順と、

前記格納手段に格納された符号データをJ P E G 2 0 0 0符号データとして出力する出力手順と

を実行させるためのプログラム。

【請求項13】 請求項12記載のプログラムを格納したことを特徴とするコンピュータ読み取り可能な記録媒体。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】

本発明は、ネットワークを介して受信した断片的な画像データから符号データを作成する技術に関する。

【0002】

【従来の技術】

インターネット上では、WWWサーバにWeb（ウェブ）ブラウザからアクセスして文書データや画像データ等の情報を閲覧することが盛んに行われている。このように、インターネット上に情報を公開するWWWサーバと、その情報を閲覧するためのクライアントとを含むシステム環境において、各クライアントではウェブブラウザを使用して、WWWサーバによって公開された情報を閲覧することができる。

【0003】

WWWサーバには、通常、ホームページといわれる公開するための情報をHTMLで記述した文書が保存されており、それにクライアント側のウェブブラウザがアクセスしてクライアント側のコンピュータに表示する。また、クライアント側のウェブブラウザは、表示しているページ内のリンクを辿っていくことにより、必要な情報を得ることができる。さらに、WWWサーバが管理しているファイルをダウンロードする方法として、File Transfer Protocol（以下、「FTP」と略す。）という方法がある。このFTPとは、ネットワークを通して、WWWサーバ上にあるファイルの内容を一度にクライアント側のコンピュータに転送する仕組みのことである。

【0004】

また、サーバが管理する画像ファイルへ断片的にアクセスして、クライアントで表示するためのプロトコルとして、Flashpix/IIPがある。このインターネット・イメージング・プロトコル（IIP）は、Flashpixという画像データファイルフォーマットに最適なプロトコルになっており、ネットワークを通して画像データの一部分を要求できるものである。このときのアクセスは、Flashpixのタイル単位に行われる。

【0005】

一方、このFlashpix/IIPをそのままJPEG2000に適用した場合を考える。ここで、JPEG2000では、各スケーラビリティ（Scalability）の符号データは、そのスケーラビリティより1つ下のスケーラビリティのデータとの差分データで構成されている。そこで、クライアント側で受信した断片的な符号データをキャッシュしておき、全符号データをデコーダに引き渡して最初から再デコードする方法と、デコーダを途中で止めておいて今回受信した符号データをデコーダに渡し、前回の続きからデコードする方法とがある。

【0006】

このような方法のうち、マルチ解像度データを有する画像符号データの中から必要な部分のデータのみを取り出して別の符号データに変換する方法が従来から提案されている（例えば、特許文献1参照）。

【0007】

上記特許文献1においてソースとなる画像データは、ウェーブレット (Wavelet) 変換、或いはWavelet-likeな変換を用いて、マルチ解像度のデータを管理することが可能な符号データである。そして、このソースとなる符号データの中から、ユーザが選択した空間的な領域のデータを処理するために必要な符号データを取り出し、1つの独立した符号データに変換する方法が記載されている。尚、この時ソースの符号データから取り出される部分的な符号データは、J P E G 2 0 0 0のコードブロックに対応しており、今回のユーザからの要求を処理するために必要な符号データを含んでいる。

【0008】

ここで、サーバから送られてくる符号データをクライアント側で最初から再デコードする場合、それらの断片的な符号データをキャッシュのような仕組みを持たないで、直接、J P E G 2 0 0 0 準拠の1つの符号データファイルに作り直すことが可能である。

【0009】

【特許文献1】

米国特許第6041143号明細書

【0010】

【発明が解決しようとする課題】

しかしながら、サーバ側から送られてくる断片的な符号データは、クライアント側から要求する順に送られてくるため、クライアント側で受信した符号データをJ P E G 2 0 0 0 準拠の1つの符号データに変換するためには、クライアント側での煩雑な処理が必要であるという問題点がある。

【0011】

また、サーバから送られてきた断片的な符号データを独自形式のキャッシュファイルとして構成し、クライアント側のJ P E G 2 0 0 0 デコーダが直接このキャッシュファイルをリードすることも可能である。しかし、この場合、J P E G 2 0 0 0 デコーダには独自のキャッシュファイルをリードするための処理が必要となり、処理が複雑になると共に、汎用のJ P E G 2 0 0 0 デコーダが使えないという問題点がある。

【 0 0 1 2 】

そこで、J P E G 2 0 0 0 符号データをパケット (packet) 単位で受信し、それらのpacketデータをキャッシュしておいて、デコーダへ符号データを渡す際に、まだ受信していないpacketデータ部分に、zero length packet (以下、「Z L P」と略す。) データを挿入し、既に受信したpacketデータとあわせて、1つのJ P E G 2 0 0 0 準拠の符号ファイルを作成することで、メインヘッダの書き換えなどの煩雑な処理を行わずに、一般的なJ P E G 2 0 0 0 デコーダで処理可能なJ P E G 2 0 0 0 ファイルを作成することが考えられる。

【 0 0 1 3 】

しかし、受信した全てのpacketデータを利用して作成された符号データは、クライアント側のアプリケーションを使っているユーザの要求した画像サイズ、画質及び画像領域等において、ユーザの望む表示形態を考慮していない。そのため、このような符号データを受け取ったデコーダでは、その一部のデータから生成されたデコード結果を利用して表示画面が作成されることがあり、キャッシュされている全ての符号データを使って1つのJ P E G 2 0 0 0 符号データファイルを作ることは、デコーダ自体の処理にも無駄な処理が生じるという問題がある。

【 0 0 1 4 】

さらに、キャッシュされている符号データが多くなるほど、デコーダへ渡す符号データを作成するときに発生するデータのコピー作業が多くなる。これは、クライアントの表示に要するまでの時間が長くなることにつながり、パフォーマンス低下という問題が生じる。特に、画像の拡大やスクロールという命令をユーザが行うことで、表示に直接必要のないキャッシュデータが多くなり、上記の問題が頻繁に起こることになる。

【 0 0 1 5 】

さらにまた、上記特許文献 1 には、以下のような問題点がある。すなわち、上記特許文献 1 に記載の技術をネットワークを通した通信である I I P にそのまま適用した場合、クライアントからの要求毎に全階層の全コードブロックの符号データを送ることになり、クライアント側で既に受信済みの符号データまでも送信することになる。また、送信する符号データのヘッダ部分は、今回要求した画像

データの情報（例えば、画像サイズ、階層情報等）に書き換えられてしまい、サーバで管理されている符号データの本来の情報を取得することができない。

【0016】

本発明は、このような事情を考慮してなされたものであり、クライアントにキャッシュされている断片化された符号データと、サーバから必要な分だけ受信した断片化された符号データとを用いて、クライアントにおいて汎用 J P E G 2 0 0 0 デコーダで使用可能な符号データであって、当該符号データのデコード及び画像データの表示処理を高速に行うことができる符号データを好適に作成することができる符号データ作成方法及び装置を提供することを目的とする。

【0017】

【課題を解決するための手段】

上記課題を解決するために、本発明は、サーバが管理する符号データのうち、断片的な第1の符号データを格納した格納手段を備えるクライアントにおいて、J P E G 2 0 0 0 符号データを作成する符号データ作成方法であって、

前記クライアントにおいて必要となる符号データと、前記格納手段に格納された前記第1の符号データとから、不足する第2の符号データを算出する算出工程と、

前記サーバに対して、算出された前記第2の符号データを要求する要求工程と、

前記サーバから、前記第2の符号データを取得する取得工程と、

取得された前記第2の符号データを前記格納手段に格納する格納工程と、

取得された前記第2の符号データのヘッダ情報を解析して、前記符号データを複数の独立した符号データに分割する分割工程と、

前記分割工程で分割された単位毎に、前記独立した符号データを構成する全ての符号データが前記格納手段に格納されているか否かを判定する判定工程と、

前記独立した符号データを構成する全ての符号データが格納されていない場合、格納されていない部分にダミー符号データを格納するダミー格納工程と、

前記格納手段に格納された符号データを J P E G 2 0 0 0 符号データとして出力する出力工程と

を有することを特徴とする。

【0018】

また、本発明に係る上記符号データ作成方法は、前記判定工程によって、前記独立した符号データを構成する全ての符号データが前記格納手段に格納されていると判定された場合、該格納手段に格納された該符号データを前記分割工程で分割された前記独立した符号データに置き換える置換工程をさらに有することを特徴とする。

【0019】

さらに、本発明は、第1のコンピュータが管理する符号データのうち、断片的な第1の符号データを格納した格納手段を備える第2のコンピュータにおける符号データ作成装置であって、

前記第1のコンピュータに管理された前記符号データのうち、断片的な第1の符号データを格納する第1の格納手段と、

前記第2のコンピュータにおいてJ P E G 2 0 0 0符号データの作成に必要な符号データと、前記格納手段に格納された前記第1の符号データとから、不足する第2の符号データを算出する算出手段と、

前記第1のコンピュータに対して、算出された前記第2の符号データを要求する要求手段と、

前記第1のコンピュータから、前記第2の符号データを取得する取得手段と、

取得された前記第2の符号データを格納する第2の格納手段と、

取得された前記第2の符号データのヘッダ情報を解析して、前記符号データを複数の独立した符号データに分割する分割手段と、

前記分割手段で分割された単位毎に、前記独立した符号データを構成する全ての符号データが前記第1又は第2の格納手段に格納されているか否かを判定する判定手段と、

前記独立した符号データを構成する全ての符号データが格納されていない場合、格納されていない部分にダミー符号データを格納する第3の格納手段と、

前記第1、第2及び第3の格納手段に格納された符号データを用いてJ P E G 2 0 0 0符号データを作成する作成手段と

を備えることを特徴とする。

【0020】

【発明の実施の形態】

以下、図面を参照して、本発明の実施形態について詳細に説明する。

【0021】

<第1の実施形態>

図1は、ネットワーク上に複数のコンピュータが接続されている様子を示す図であり、本発明の第1の実施形態に係るサーバ及びクライアントを含むネットワークシステムの構成を示す概要図である。

【0022】

図1において、100はインターネットに代表されるネットワークである。101、103は、ネットワーク100に接続されているサーバ・コンピュータであり、画像データを送信するためのJ P E G 2 0 0 0用のI I Pサーバを始めとして、WWWサーバ機能に必要なソフトウェアが実行されている。また、大量の画像データも格納されている。また、102a、102bはクライアント・コンピュータであり、ウェブブラウザやJ P E G 2 0 0 0デコーダ等のクライアント側で必要なソフトウェアが実行されている。

【0023】

図2は、図1のサーバ・コンピュータ101、103又はクライアント・コンピュータ102a、102bの各コンピュータシステムのハードウェア構成の一例を示す図である。図2において、201はC P Uであり、システム全体の制御等を行っている。202はキーボードであり、マウス201aと共に本システムに対して情報等を入力するために使用される。また、203は表示部であり、C R Tや液晶ディスプレイ等で構成されている。

【0024】

一方、204はR O M、205はR A Mであり、本システムにおける記憶装置を構成しており、システムが実行するプログラムやシステムが利用するデータ等を記憶する。また、206はハードディスク装置、207はフレキシブルディスクであり、本システムのファイルシステムに使用される外部記憶装置を構成して

いる。さらに、208はプリンタである。さらにまた、209はネットワークを制御する部分であり、ここからネットワーク100（例えば、インターネット等）上に接続されているサーバ・コンピュータ等のネットワーク上の資源にアクセスする。

【0025】

本実施形態では、既に生成済みのJ P E G 2 0 0 0ファイルがサーバ・コンピュータで管理されている。一方、クライアント・コンピュータでは、画像表示アプリケーション等を用いてユーザが要求したJ P E G 2 0 0 0ファイル内の必要な部分のみを、J P E G 2 0 0 0用I I Pを使用して上記サーバ・コンピュータにアクセスし、符号データを断片的に受信し、受信した断片的な符号データをクライアント・コンピュータ上の例えばハードディスク206等にキャッシュする。

【0026】

以下では、ハードディスク206等にキャッシュされた断片的なJ P E G 2 0 0 0符号データを、J P E G 2 0 0 0のデコーダに渡すJ P E G 2 0 0 0準拠の符号データに変換する部分について説明する。

【0027】

また、ユーザは、Windows（登録商標）マシン等を使用してホームページを開き、そこに書かれているJ P E G 2 0 0 0の画像へのリンクをクリックすることで、J P E G 2 0 0 0の画像をユーザの用途に適した画像サイズや解像度で表示するために必要な断片的なデータを取得し、キャッシュする。そして、それらのキャッシュデータから一つのビットストリームを作り出し、デコードし、表示するという場合を想定する。

【0028】

ここで、一般的なJ P E G 2 0 0 0の符号データについて説明する。図3は、Layer-resolution level-component-position progression（以下、「S N R Pr ogression」と記す。）に沿って記録されたJ P E G 2 0 0 0ファイルの構成を示す概念図である。S N R Progressionに準じた場合、layer/resolution/component/positionの順に記録される。このような符号データの並び方を規定する

ことは、「progression order」と呼ばれる。

【0029】

また、図4は、J P E G 2 0 0 0の解像度スケーラビリティを説明する図である。すなわち、解像度（画像サイズ）とResolution番号との関係を示すための図である。図4では、最も小さい解像度の画像のresolution番号を0とし、resolution番号が1つ増加するごとに画像の幅と高さが2倍になっていく。また、各layer内は、resolution番号の小さい順にデータが格納されている。そして、Layer番号は、復元する画像の原画に対するS/N比に対応しており、layer番号が小さいほどS/N比が悪くなる。

【0030】

さらに、1つのJ P E G 2 0 0 0ファイル内でのresolution番号、layer番号及びcomponent番号の最大値は、エンコーダによって予め設定されており、そのパラメータに従ってエンコードされており、その情報は符号データの中に格納されている。また各packetは、そのpacket内に格納されているcode-blockの情報を管理しているpacket header部と、各code-blockの符号データとから構成されている。

【0031】

このようなJ P E G 2 0 0 0符号データを使うことによって、ユーザは、サーバにある全ての画像データを取得する必要がなく、クライアント側で表示等するために必要な部分の符号データのみをサーバから受信すればよい。尚、ユーザ側（クライアント・コンピュータ）の受信データの単位としては、J P E G 2 0 0 0のpacket、或いはpacketよりも更に小さい符号化単位であるコードブロック単位が考えられる。本実施形態では、ユーザがサーバから受信するデータ単位としてpacket単位を想定する。

【0032】

図5は、第1の実施形態におけるサーバとクライアント間でのpacket単位のデータのリクエスト及びレスポンスを説明するための概念図である。

【0033】

図5において、クライアント501はサーバ502に画像のタイル番号、reso

lution levelと、layer、component、position番号を指定して必要なデータを要求する。そして、サーバ502は、画像503のコードストリームを解析して、指定された画像のタイル番号、resolution levelとlayer、component、position番号に相当するpacketデータを抜き出してクライアント501に送り返す。

【0034】

次に、図6A及び図6Bを用いて本実施形態で使用するサーバ・コンピュータで管理されているJ P E G 2 0 0 0ファイルの一例について説明する。図6Aは、サーバ・コンピュータに格納されている原画像データをタイルに分割した例を示す図である。図6Aでは、縦横とも各々4つのタイル、すなわち画像全体で16個のタイルに分割した例が示されている。

【0035】

また、図6Bは、図6Aで示したタイル分割された画像データに関するJ P E G 2 0 0 0符号データ（すなわち、当該画像データを「S N R Progression」でエンコードしたデータ）のデータ構造を示す図である。図6Bに示される符号データは、layerは0～2の3種類（Lay0、Lay1、Lay2）、resolution levelは、0～3の4種類（Res0、Res1、Res2、Res3）、componentとpositionは各1種類（Com0、Pos0）である場合のJ P E G 2 0 0 0で符号化した時の符号データの構成例である。

【0036】

例えば、J P E G 2 0 0 0の最大解像度の画像サイズを1024×1024画素とした場合、各タイルの画像サイズは256×256画素となる。また、各タイルが4種類のresolution levelを持つ場合の各解像度での画像サイズは、256×256、128×128、64×64、32×32画素となる。

【0037】

さらに、J P E G 2 0 0 0符号データ内のメインヘッダ部分については、図6Bに示すように、画像サイズを示すパラメータであるXsiz、Ysizが共に1024であり、画像全体の画像サイズを示している。一方、タイルのサイズを示すXTsiz、YTsizは、上述したタイルのサイズである256の値を持っている。これらのパラメータにより、このJ P E G 2 0 0 0符号データは、16個のタイル

に分割された符号データを有することになる。そして、これらの各タイルの符号データが、図 6 B に示すように「Tile part Header」で区切られて格納されている。

【0038】

図 7 は、第 1 の実施形態におけるクライアント・コンピュータ上のアプリケーションでの処理手順の概要を説明するためのフローチャートである。本実施形態においては、クライアント側では、J P E G 2 0 0 0 画像表示アプリケーションが起動しており、図 7 に示されるフローチャートは、表示したい J P E G 2 0 0 0 ファイルをウェブ等の別の手段で指定した後、画像を表示する部分の処理フローである。

【0039】

まず、これからの処理を制御するための内部変数（パラメータ）を初期化する（ステップ S 7 0 7）。具体的には、指定されている J P E G 2 0 0 0 ファイルの画像サイズや、エンコード・パラメータ等の情報をサーバに対して問い合わせ、図 1 3 に示すこれから表示する符号化データのキャッシュを管理するための情報をメモリやファイル上に作成し、その変数（パラメータ）等を初期化する。すなわち、図 1 3 は、クライアント側で制御している符号データのキャッシュの論理構造を説明するための図である。図 1 3 において、1 3 0 0 は、キャッシュファイル全体の論理構造であり、図に示すように、キャッシュ全体に関する画像管理情報フィールド 1 3 0 2、Main header data フィールド 1 3 0 3、各 Tile に関するデータフィールド 1 3 0 1 に分けることができる。

【0040】

ここで、Tile に関するデータフィールド 1 3 0 1 には、この画像データを Tile 分割した時の Tile の個数分のフィールドが存在する。1 3 0 4 は、Tile に関するデータのフィールド 1 3 0 1 の詳細を示したものである。この中には、Tile データ管理用データフィールド 1 3 0 4 a、Tile header フィールド 1 3 0 4 b、packet データ管理用データ 1 3 0 4 c に分けられる。尚、packet データ管理用データフィールド 1 3 0 4 c は、Tile 内に存在する packet の個数分のフィールドがある。

【0041】

次に、Tileデータ管理用データフィールド1304aの詳細を1305に示す。Tileデータ管理用データフィールド1304aは、1305に示すように、Tileのデータ長1305a、Tile番号1305b、packet数1305c、Resolution数1305d、Layer数1305e、Component数1305f、Position数1305gの各パラメータから成る。

【0042】

さらに、packetデータの管理用データ1304cの詳細を1306に示す。packetデータの管理用データ1304cは、1306に示すように、packetのデータ長1306a、ポインタ値1306b、packet番号1306c、packetの基本情報1306dから成る。サーバから受信した実際のpacketのデータは、ポインタ値1306bで示すポインタの先に保存される。

【0043】

図13からも分かるように、packetデータをフル充填した場合は、テーブルを管理するための情報等が数多く存在するため、サーバで管理している元の符号データのバイト数よりも大きくなってしまう。すなわち、クライアント側では、このキャッシュ情報と、キャッシュ情報から1つの符号データを生成するため、クライアント側で管理する符号データ関連のバイト数は、全タイルがフル充填されたときに最大となり、その時のサイズは、サーバで管理している元の符号データのサイズの2倍以上になる。尚、これらの各Tileに関するデータ1304は、ファイルに格納され、1300内の1301で示すフィールドには、ファイル名で該当するデータと関連付けを行っているものとする。

【0044】

ステップS707で内部変数が初期化された後、ユーザが表示のための操作を行う（ステップS700）。例えば、アプリケーション起動時の指定されたファイルの1回目の表示では、開いたウィンドウサイズに収まる画像サイズを計算して、当該ウィンドウサイズに一致する画像サイズで表示するようにする。そして、その後に本ステップS700でユーザの操作を受け付けるようにしてもよい。

【0045】



次に、上記ステップ S 7 0 0 でのユーザ操作により新たに必要になった packet を全て算出する（ステップ S 7 0 1）。その後、ステップ S 7 0 1 で算出した新規に必要な packet のデータについてサーバに対して要求を発行する（ステップ S 7 0 2）。次いで、要求した全 packet のデータをサーバから受信し、クライアント・コンピュータ上にキャッシュする（ステップ S 7 0 3）。

【0046】

さらに、高速読み出しが可能なようにキャッシュ形式で管理している符号データを複数の J P E G 2 0 0 0 符号データ形式に変換することによって符号データの作成を行う（ステップ S 7 0 4）。この符号データの作成処理の詳細については後述する。その後、ステップ S 7 0 4 で作成した J P E G 2 0 0 0 符号データをデコードして表示する（ステップ S 7 0 5）。そして、ユーザが終了を要求したか否かを判定する（ステップ S 7 0 6）。その結果、ユーザが終了を要求した場合（Y e s）、本アプリケーションを終了する。一方、ユーザが終了を要求せずに他の表示要求を行った場合（N o）、ステップ S 7 0 1 に戻って上記処理を実行する。

【0047】

以下では、上記クライアント・コンピュータ上のアプリケーションでの処理手順の具体的な処理内容について説明する。

【0048】

まず、クライアント・アプリケーションが起動した直後で、このアプリケーションの初期画像表示サイズを 128×128 画素とし、このサイズに入る画像データを表示することを考える。この場合、ステップ S 7 0 2 で算出される必要な packet は、最低解像度の画像サイズであって S N R が最大のデータを表示することを考えると、図 6 B 内のタイル 0 からタイル 1 5 までの全 1 6 タイルにおける、「Lay0/Res0/Com0/Pos0」、「Lay1/Res0/Com0/Pos0」、「Lay2/Res0/Com0/Pos0」で示す 3×16 個の packet が必要な packet である。

【0049】

そこで、クライアント・プログラムは、これらの packet をサーバ・プログラムに要求する。一方、サーバ・プログラムは、この全タイルの 3×16 個の packet



を切り出す処理を行う。その後、クライアント・プログラムは、これらのpacketを受信した後デコードして表示を行う。

【0050】

次に、図7のステップS704における符号データ作成についての詳細な説明を図8～10を用いて行う。図8Aは、図7におけるステップS704符号データ作成処理の概要を説明するためのフローチャートである。まず、サーバから受信した必要なpacketに関するJ P E G 2 0 0 0符号データのメインヘッダを解析する（ステップS800）。次に、ステップS800におけるメインヘッダを解析した結果から、符号データ内のタイルの個数を計算する（ステップS801）。本実施形態においては、タイルの個数の計算は、以下に示す式（1）で行うものとする。

【0051】

$$\begin{aligned} Xtiles &= (Xsiz + XTsiz - 1) \div XTsiz \\ Ytiles &= (Ysiz + YTtiz - 1) \div YTtiz \\ Tiles &= Xtiles \times Ytiles \end{aligned} \quad (1)$$

そして、式（1）で求められたタイルの個数だけ、ステップS802以下の処理が行われる。

【0052】

ステップS802は、各タイルの符号変換処理を行う部分である。この処理の詳細については後述する。そして、全タイルについての符号変換処理が終了したか否かを判断する（ステップS803）。その結果、未処理のタイルがある場合（N o）、ステップS802に戻る。一方、全タイルについて符号変換処理が終了した場合（Y e s）、本処理を終了してステップS705に進む。

【0053】

図8Bは、ステップS802における符号変換処理を詳細に説明するためのフローチャートである。まず、今回処理するタイルがフル充填、すなわち全packetデータがキャッシュされているか否かを判断する（ステップS804）。具体的には、図13に示すpacketのデータ長1306a、或いはpacketデータへのポインタ値1306bを見ることで判断できる。これらの値は、未受信の場合は「0

」であり、packetデータを受信すると、packetデータのバイト数や、実際に格納されているアドレス或いはファイル名等の各値が格納される。これにより、このタイル内の全てのpacketに対して、データ長1306a或いはポインタ値1306bの値を判断し、各値が「0」でなければフル充填であると判断できる。逆に、1つでも「0」の値が存在すれば、まだフル充填ではないと判断することができる。その結果、フル充填されている場合（Yes）、ステップS805に進み、それ以外の場合（No）、ステップS807に進む。

【0054】

ステップS805では、フル充填状態の符号データを作成したか否かを判断する。その結果、まだ作成されていない場合（No）はステップS806に進み、作成済みの場合（Yes）は当該符号変換処理を終了する。ステップS806では、完全符号データを作成したことを示すフラグをセットする。当該フラグのセットに係る処理の詳細については後述する。そして、実際のキャッシュデータから符号データを作成する「符号ファイル作成」処理が行われる（ステップS807）。この符号ファイル作成処理の詳細については後述する。

【0055】

ここで、図8のステップS806の詳細な処理内容について説明する。図14は、図8のステップS806の完全符号データ作成済を示すフラグのセットを詳細に説明するためのフローチャートである。この処理は比較的容易であり、まず、フル充填されたタイルのTileデータ管理用データ1304のファイルを削除する（ステップS1400）。これにより、図13で示すTileに関するデータ1304、Tileデータの管理用データ1305、Packetデータの管理用データ1306の領域が削除される。次に、図11等で示すタイル毎の符号データファイルに置き換えて完全符号データをセットする（ステップS1401）。すなわち、完全符号データ作成済を示すフラグのセットとは、図13に示す1300内の1301で示すTileに関するデータの値を変更することである。

【0056】

上記処理を行うことにより、当該処理後にクライアントのアプリケーションがこのタイルのデータを必要とするときは、1301で示すフィールドに格納され

ているファイル名をアプリケーションに渡すだけでよく、データ変換等の処理は全く発生せず、高速に行うことができる。さらに、キャッシュとして保持しなければならないデータ量が、管理テーブルの形で保持する場合よりも符号データを直接持つことになるため、データ容量的に小さくて済む。

【0057】

また、タイルがフル充填されているか否かの判断は、キャッシュとして保持されているTileのデータ長のフィールド1305aを確認すればよい。図15は、タイルがフル充填されて完全な符号データに置き換わった時の状態を示す図である。具体的な確認方法は、図15に示されるフィールドを16ビット長の整数のフィールドとする場合、フル充填されて置き換えた場合は、符号データの先頭16ビットになる。この場合、符号データの先頭は、SOCマーカ・コード(0xFF4F)であり、整数でこの値を読むと先頭のビット(15ビット目)が"1"であるので、負の値になる。32ビット長の整数としても結果は同じであり、SOCマーカ・コード、SIZマーカ・コードであるので、(0xFF4FFF51)である。すなわち、Tileデータの管理用データの先頭を必要なビット数の整数としてリードした場合、正の整数の場合はまだ充填中であり、負の整数の場合は、フル充填で符号データに変換済みであると容易に判断することができる。

【0058】

上述したように、1度フル充填されたキャッシュデータから符号ファイルを生じた後は、再度符号ファイル作成処理(ステップS807)を行わないように制御することにより、キャッシュが充填されて行くに連れて、符号ファイル作成処理を高速に行うことができる。

【0059】

図9は、図8に示すステップS807の符号ファイル作成処理を詳細に説明するためのフローチャートである。まず、タイルに含まれるpacketの数をiPMaxにセットする(ステップS900)。次に、「Tile part Header」を仮出力する(ステップS901)。ここで仮出力するのは、この「Tile part Header」にはタイルの符号長を格納するフィールドがあり、このタイルの符号長は以下の処理を終了した時点で計算できる値であるためである。

【0060】

次に、変数*iP*、*iL*に初期値として「0」をセットする（ステップS902、S903）。ここで、変数*iP*はpacketのIDを示す変数であり、変数*iL*はタイルの符号長を示す変数である。次に、変数*iP*で示されるpacketがキャッシュされているか否かを判断する（ステップS904）。その結果、変数*iP*がキャッシュされている場合（Yes）はステップS905へ進み、変数*iP*がキャッシュされていない場合（No）はステップS907に進む。

【0061】

ステップS907では、該当するpacketデータが無いことを示す「Zero Length Packet」のコード（以下、「ZLPコード」と略す。）を出力する。そして、このZLPコードのバイト数である「1」を変数*iL*に加える（ステップS908）。一方、ステップS905では該当するpacketのデータを出力する。そして、そのバイト数を変数*iL*に足し込んで更新する（ステップS906）。

【0062】

ステップS906及びS908の処理の後、変数*iP*をインクリメントする（ステップS909）。そして、全packetについて処理したかを判断する（ステップS910）。その結果、まだ全packetを処理していない場合（No）、ステップS904に戻って上記処理を行う。一方、全packetを処理した場合（Yes）、変数*iL*から「Tile part Header」である「SOTマーカ」を更新して出力し（ステップS911）、当該符号ファイル生成処理を終了する。そして、前述したように当該符号ファイルを汎用のJPEG2000デコーダでデコードし、デコードされた画像データを表示する（ステップS705）。

【0063】

すなわち、上述したように、本実施形態に係るクライアントは、サーバが管理する符号データのうち、断片的な第1の符号データを格納した格納手段（例えば、ハードディスク206）を備え、JPEG2000符号データを作成する符号データ作成装置として機能する。本実施形態に係るクライアントでは、まず、当該クライアントにおいてJPEG2000符号データの作成に必要な符号データと、予めハードディスク206等の格納手段に格納された第1の符号データ

(例えば、現在表示部 203 に表示されている画像データについての符号データ) とから、不足する第 2 の符号データ (例えば、表示部 203 で拡大表示するために必要な画像データについての符号データであって、第 1 の符号データには含まれていないもの) を算出する。そして、サーバに対して、算出された第 2 の符号データを要求し、サーバから、第 2 の符号データを取得し、取得された第 2 の符号データをハードディスク 206 等の格納手段に格納する。次いで、取得された第 2 の符号データのヘッダ情報を解析して、対象となる符号データを複数の独立した符号データに分割する。さらに、分割された単位毎に、独立した符号データを構成する全ての符号データがハードディスク 206 等の格納手段に格納されているか否かを判定する。そして、独立した符号データを構成する全ての符号データが格納されていない場合、格納されていない部分にダミー符号データを格納し、格納された符号データを J P E G 2 0 0 0 符号データとして出力することを特徴とする。

【0064】

また、上記クライアント (符号データ作成装置) では、符号データが、パケット単位で取り扱われることを特徴とする。さらに、上記クライアント (符号データ作成装置) では、ダミー符号データが、J P E G 2 0 0 0 で規定されている zero length packet データであることを特徴とする。さらにまた、上記サーバとクライアントが、ネットワークを介して互いに通信可能であることを特徴とする。

【0065】

さらに、上記クライアント (符号データ作成装置) は、画像データを表示する表示手段 (例えば、表示部 203) をさらに備え、上記第 1 の符号データが当該画像データの符号データであって、表示部 203 等の表示手段に表示された画像データの表示領域の移動又は拡大表示に応じて、上記第 2 の符号データの算出において必要となる符号データを設定する。また、上記クライアント (符号データ作成装置) は、出力された上記 J P E G 2 0 0 0 符号データをしてデコードし、デコードされた画像データを表示部 203 等の表示手段に画面表示することを特徴とする。

【0066】

ここで、具体例を用いて、上記フローチャートで示した符号ファイル作成処理を説明する。図10は、本発明の第1の実施形態における初期画面での符号ファイル処理後の各タイルの符号ファイルの構成を示す図である。ここで、クライアント・コンピュータから要求されるpacketは、前述のとおり全タイルの「Lay0/Res0/Com0/Pos0」、「Lay1/Res0/Com0/Pos0」、「Lay2/Res0/Com0/Pos0」である。

【0067】

従って、図10に示すように、全16個のタイルに対して、符号1001で示すように上記3つのpacketデータ以外のpacketについてはZLPが格納された符号データが作成される。また、各タイル毎に作成される符号データのメインヘッダ部分は、メインヘッダ内のSIZマークコード内のXsiz、Ysizの各フィールドがタイルのサイズである256に変更されている。これにより、各タイル毎に独立した符号ファイルを作成することができる。

【0068】

すなわち、本実施形態に係るクライアント（符号データ作成装置）では、符号データ（画像データ）を所定サイズのタイル単位（例えば、式（1）に記載の方法で算出された単位）で分割することを特徴とする。また、同クライアントでは、独立した符号データのそれぞれのヘッダ情報を、分割されたタイル単位のサイズに変更することを特徴とする。

【0069】

次に、ステップS701でユーザから拡大表示が指示された場合について説明する。この拡大表示では、まず初期画面より2倍に拡大するものとする。この場合、表示ウィンドウサイズは128×128画素であるとする、表示できるタイルは4個になる。例えば、図6Aに示すTile5、Tile6、Tile9、Tile10の4個のタイルのResolution1（画像サイズ：64×64画素）を表示するものとする。この場合、ステップS702では、Tile5、Tile6、Tile9、Tile10の4つのタイルの「Lay0/Res1/Com0/Pos0」、「Lay1/Res1/Com0/Pos0」、「Lay2/Res1/Com0/Pos0」がクライアント・コンピュータからサーバに対して要求される。

【0070】

さらに、ユーザから2倍に拡大する要求が発行された場合、表示ウィンドウサイズを128×128画素から256×256画素に変更すると、上記4タイルに対してさらに、「Lay0/Res2/Com0/Pos0」、「Lay1/Res2/Com0/Pos0」、「Lay2/Res2/Com0/Pos0」が要求される。

【0071】

その後、さらに、2倍に拡大する要求がユーザからあったとする。この場合、表示領域のウィンドウサイズを変更しないとすると、タイルが1つ分しか表示することができない。そこで、Tile5のみを表示するとすると、Tile5の「Lay0/Res3/Com0/Pos0」、「Lay1/Res3/Com0/Pos0」、「Lay2/Res3/Com0/Pos0」をサーバに要求し表示することになる。これにより、Tile5は完全にキャッシュが充填されたことになる。図11は、第1の実施形態における符号データの各タイル毎の符号ファイルの状態の一例を説明するための図である。

【0072】

図11に示すように、Tile0、Tile1、Tile2、Tile3、Tile4、Tile7、Tile8、Tile11、Tile12、Tile13、Tile14及びTile15の各タイルの符号ファイルは、図10に示したものと同様に符号1001で示す状態になっている。また、Tile6、Tile9及びTile10の各タイルは符号1101に示す状態に、Tile5は符号1102に示す状態になっている。この状態になるまでは、ステップS805における完全符号データを作成済みか否かの判断により全てステップS806に進むようになるが、Tile5の符号変換処理後はステップS805の判断後、ステップS806には進まずに処理を終了するようになる。これにより、Tile5については、これ以降、符号変換処理を行う必要がなくなる。これにより、キャッシュが充填されるほど符号ファイル作成処理を高速にすることができる。すなわち、本実施形態によれば、デコード／表示する領域をタイル単位で行う場合、デコード／表示に必要なタイルのみに対して処理を行うことができる。

【0073】

また、従来は、キャッシュファイルから1つのJPEG2000符号データファイルを生成する場合、表示するタイル毎のマルチスレッド化は困難であった。これは、1つの符号データファイルを生成するため、ここで処理をシリアルライズ

することが必須であったためである。しかし、本実施形態のように各タイル毎の符号データファイルを生成することにより、表示するタイル毎に必要なpacket要求からキャッシュファイルの生成、キャッシュファイルから符号データの作成、符号データのデコード及び表示までを各タイル毎スレッドとしてマルチスレッド化することができる。従って、これまでのシングルスレッドの場合に比べて、マルチスレッド化することによる高速化も図ることができる。

【0074】

さらに、ファイル I/O の処理時間がかかるようなシステムにおいては、1つの大きな符号データファイルを作成する代わりに、小さな複数の符号ファイルを生成することで、ファイル I/O による時間のロスを最小限にすることも可能となる。

【0075】

<第2の実施形態>

上述した第1の実施形態では、キャッシュされている全てのタイルに対して符号変換処理を行うことを説明したが、本実施形態では、表示に必要な部分のタイルのみ符号変換処理を行うことにより、第1の実施形態よりもさらに高速に符号変換処理を行う場合について説明する。

【0076】

図12は、本発明の第2の実施形態に係る符号データ作成処理の概要を説明するためのフローチャートである。尚、上述した第1の実施形態における図8Aのフローチャートと同じ処理の部分には同一の番号を付けている。ここでは、第1の実施形態と異なる部分のみを説明する。

【0077】

ステップS801のタイル個数の計算の後、ステップS701でユーザが操作したコマンドを解析する（ステップS1200）。そして、この解析結果を使用して、これから処理しようとしているタイルが今回の表示で表示されるタイルか否かを判断する（ステップS1201）。その結果、ここで表示に使用しないタイルであると判断された場合（No）、ステップS803に進む。一方、表示に使用するタイルであると判断された場合（Yes）、ステップS1202に進む

。

【0078】

ステップS1202では、ステップS703で新たにpacketを要求した場合に、今回処理しようとしているタイルのpacketを取得したか否かを判断する。その結果、新規packetを取得している場合（Yes）はステップS802に進み、新規packetを取得していない場合（No）はステップS803に進む。尚、これ以降の処理手順については、第1の実施形態で説明した手順と同様である。これらの処理により、表示に必要なタイルであって、まだ完全に充填されていないタイルのみ符号変換処理を行うことができるため、当該符号化処理を第1の実施形態よりも高速に行うことができる。

【0079】

<第3の実施形態>

以下、本発明の第3の実施形態に係る画像データ送信装置について、図面を参照して、詳細に説明する。本実施形態では、キャッシュから1本の符号データを作成する方法について説明する。尚、本実施形態に係るサーバ及びクライアントを含むネットワークシステムの構成は図1と同様であり、各コンピュータシステムのハードウェア構成も図2と同様である。また、クライアントコンピュータ上のアプリケーションでの処理手順についても図7に示すフローチャートと同様である。

【0080】

図16は、キャッシュから1本の符号データファイルを作成する場合の処理手順を説明するためのフローチャートである。図16に示すように、まず、キャッシュ内のフィールド1303で保持している情報をJPEG2000符号データ形式に則った形式に変換し、出力ファイルにメインヘッダ（Main Header）を出力する（ステップS1600）。次に、メインヘッダを解析した結果から、符号データ内のタイルの個数を計算する（ステップS1601）。尚、この際の計算式は、図8のステップS801の場合と同様である。そして、以下では、前述した式（1）で求められたタイルの個数だけ、ステップS1602以下の処理が行われる。

【0081】

そして、各タイルごとに、当該タイルがフル充填しており、タイルの完全な符号データがあるか否かを判断する（ステップS1602）。尚、この判断方法は、前述の通りである。その結果、フル充填されていると判断された場合（Yes）、ステップS1604へ進み、フル充填されていない場合（No）、ステップS1603へ進む。

【0082】

ステップ1603では、図9に示す符号ファイルの作成処理手順と同様にして符号ファイルを作成する。そして、この処理後はステップS1606へ移り、符号データを出力する。一方、ステップS1604以降では、タイルの完全な符号データからの変換が行われる。そこで、まず、メインヘッダ部分を読み飛ばす処理が行われる（ステップS1604）。この処理の具体例としては、例えば、図15に示す符号データの先頭からSOTマーカ・コードまでを読み飛ばせばよい。次に、タイルパートヘッダ（Tile part Header）内のタイルインデックス（Tile Index）の値を当該タイルのインデックス（Index）番号に置き換える（ステップS1605）。さらに、タイルパートヘッダ以降EOCマーカ・コードまで（但し、EOCマーカ・コードは含まない）をコピーする（ステップS1606）。そして、全タイルを処理したかどうかを判断し（ステップS1607）、未処理の場合（No）はステップS1602に戻って上記処理を繰り返す。一方、全タイルが処理された場合（Yes）は終了する。

【0083】

上述した処理によって、本実施形態に係る形式でのキャッシュファイルから、1本の符号データを簡単に、且つ、データ変換等の処理がないため高速に作成することができる。

【0084】

<その他の実施形態>

一般に、キャッシュファイルから1つのJPEG2000符号データファイルを生成する場合には、表示するタイル毎のマルチスレッド化は困難であった。なぜならば、1つの符号データファイルを生成するために、処理をシリアルライズす

ることが必須であったからである。しかし、上述したように、各タイル毎の符号データファイルを生成することにより、表示のタイル毎に必要なpacket要求から、キャッシュファイルの生成、キャッシュファイルからの符号データの作成、符号データのデコード、表示までを各タイル毎スレッドとしてマルチスレッド化を行うことができる。そのため、これまでのシングルスレッドの場合に比べて、マルチスレッド化することによる高速化も図ることができる。

【0085】

尚、本発明は、複数の機器（例えば、ホストコンピュータ、インタフェース機器、リーダ、プリンタ等）から構成されるシステムに適用しても、一つの機器からなる装置（例えば、複写機、ファクシミリ装置等）に適用してもよい。

【0086】

また、本発明の目的は、前述した実施形態の機能を実現するソフトウェアのプログラムコードを記録した記録媒体（または記憶媒体）を、システムあるいは装置に供給し、そのシステムあるいは装置のコンピュータ（またはCPUやMPU）が記録媒体に格納されたプログラムコードを読み出し実行することによっても、達成されることは言うまでもない。この場合、記録媒体から読み出されたプログラムコード自体が前述した実施形態の機能を実現することになり、そのプログラムコードを記録した記録媒体は本発明を構成することになる。また、コンピュータが読み出したプログラムコードを実行することにより、前述した実施形態の機能が実現されるだけでなく、そのプログラムコードの指示に基づき、コンピュータ上で稼働しているオペレーティングシステム（OS）などが実際の処理の一部または全部を行い、その処理によって前述した実施形態の機能が実現される場合も含まれることは言うまでもない。

【0087】

さらに、記録媒体から読み出されたプログラムコードが、コンピュータに挿入された機能拡張カードやコンピュータに接続された機能拡張ユニットに備わるメモリに書き込まれた後、そのプログラムコードの指示に基づき、その機能拡張カードや機能拡張ユニットに備わるCPUなどが実際の処理の一部または全部を行い、その処理によって前述した実施形態の機能が実現される場合も含まれること

は言うまでもない。

【0088】

本発明を上記記録媒体に適用する場合、その記録媒体には、先に説明したフローチャートに対応するプログラムコードが格納されることになる。

【0089】

以上説明したように、本発明による上記実施形態で説明したファイル作成方法を用いれば、IIPを用いて送られてくるJPEG2000符号化データの断片的なデータをキャッシュし、このキャッシュファイルから通常のJPEG2000デコーダで処理できるJPEG2000符号化データを作る時に、今回のデコード及び表示に必要なタイル毎に独立したJPEG2000符号データファイルを生成することで、デコード／表示までの処理を高速に行うことができる。

【0090】

また、キャッシュを構成するための、或いはキャッシュから符号データを生成するためのデータ容量或いはファイル容量を最小限に抑えることが可能になる。さらに、キャッシュから符号データを生成する時も、データ変換等の処理を省略することができるため、高速に生成することができる。

【0091】

また、デコード／表示する領域をタイル単位で行う時、デコード／表示に必要なタイルのみに対して処理を行うことができる。さらに、マルチスレッド化による高速化も行うことができる。さらにまた、ファイルI/Oの処理時間がかかるようなシステムにおいては、1つの大きな符号データファイルを作る代わりに、小さな複数の符号ファイルを生成することで、ファイルI/Oによる時間のロスをも最小限にすることも可能となる。あるいは、1つの符号データを作成するときでも、容易に変換を行うことができる。

【0092】

【発明の効果】

以上説明したように、本発明によれば、クライアントにキャッシュされている断片化された符号データと、サーバから必要な分だけ受信した断片化された符号データとを用いて、クライアントにおいて汎用JPEG2000デコーダで使用

可能な符号データであって、当該符号データのデコード及び画像データの表示処理を高速に行うことができる符号データを好適に作成することができる。

【図面の簡単な説明】

【図 1】

本発明の第 1 の実施形態に係るサーバ及びクライアントを含むネットワークシステムの構成を示す概要図である。

【図 2】

図 1 のサーバ・コンピュータ 101、103 又はクライアント・コンピュータ 102 a、102 b の各コンピュータシステムのハードウェア構成の一例を示す図である。

【図 3】

Layer-resolution level-component-position progression (SNR Progression) に沿って記録された J P E G 2 0 0 0 ファイルの構成を示す概念図である。

【図 4】

J P E G 2 0 0 0 の解像度スケーラビリティを説明する図である。

【図 5】

第 1 の実施形態におけるサーバとクライアント間での packet 単位のデータのリクエスト及びレスポンスを説明するための概念図である。

【図 6 A】

サーバ・コンピュータに格納されている原画像データをタイルに分割した例を示す図である。

【図 6 B】

図 6 A で示したタイル分割された画像データに関する J P E G 2 0 0 0 符号データのデータ構造を示す図である。

【図 7】

第 1 の実施形態におけるクライアント・コンピュータ上のアプリケーションでの処理手順の概要を説明するためのフローチャートである。

【図 8 A】

図7におけるステップS704符号データ作成処理の概要を説明するためのフローチャートである。

【図8B】

ステップS802における符号変換処理を詳細に説明するためのフローチャートである。

【図9】

図8に示すステップS807の符号ファイル作成処理を詳細に説明するためのフローチャートである。

【図10】

本発明の第1の実施形態における初期画面での符号ファイル処理後の各タイルの符号ファイルの構成を示す図である。

【図11】

第1の実施形態における符号データの各タイル毎の符号ファイルの状態の一例を説明するための図である。

【図12】

本発明の第2の実施形態に係る符号データ作成処理の概要を説明するためのフローチャートである。

【図13】

クライアント側で制御している符号データのキャッシュの論理構造を説明するための図である。

【図14】

図8のステップS806の完全符号データ作成済を示すフラグのセットを詳細に説明するためのフローチャートである。

【図15】

タイルがフル充填されて完全な符号データに置き換わった時の状態を示す図である。

【図16】

キャッシュから1本の符号データファイルを作成する場合の処理手順を説明するためのフローチャートである。

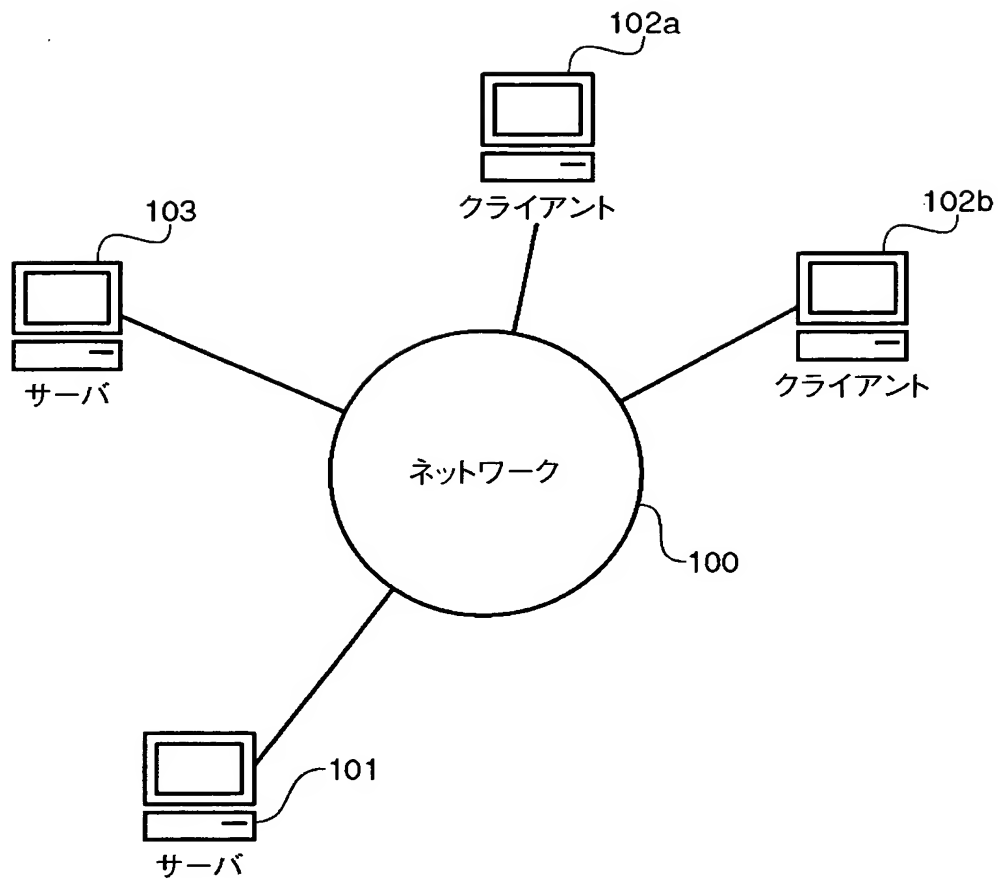
【符号の説明】

5 0 1 サーバ

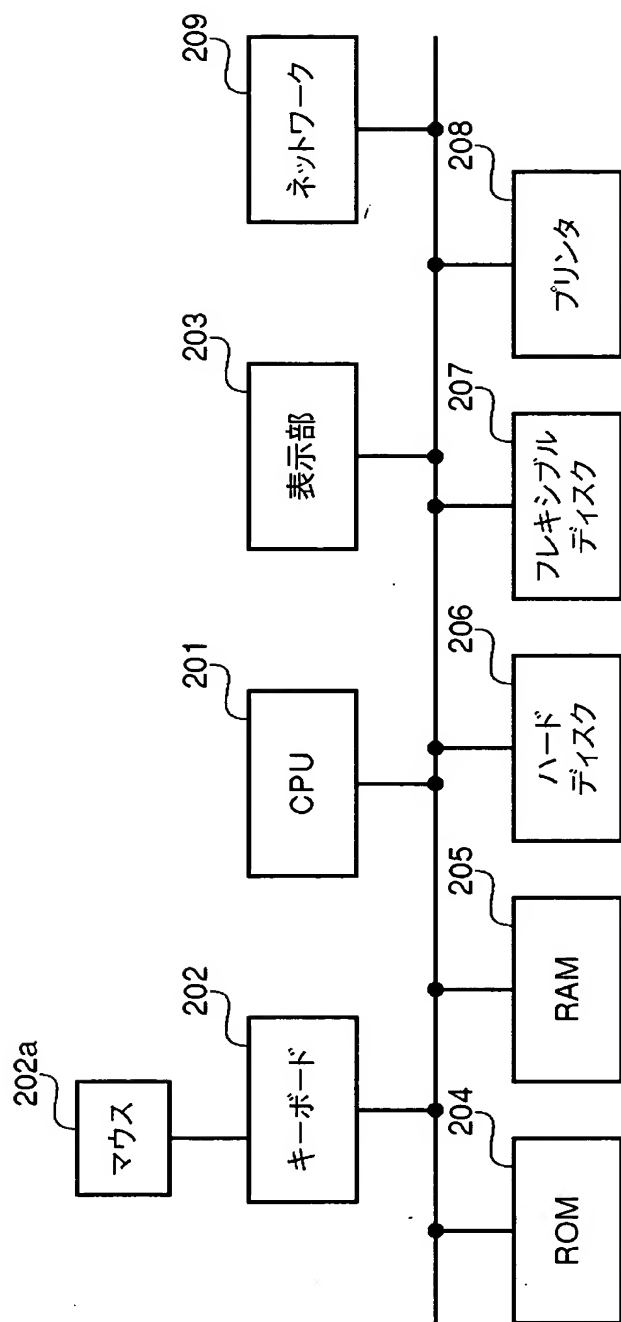
5 0 2 クライアント

【書類名】 図面

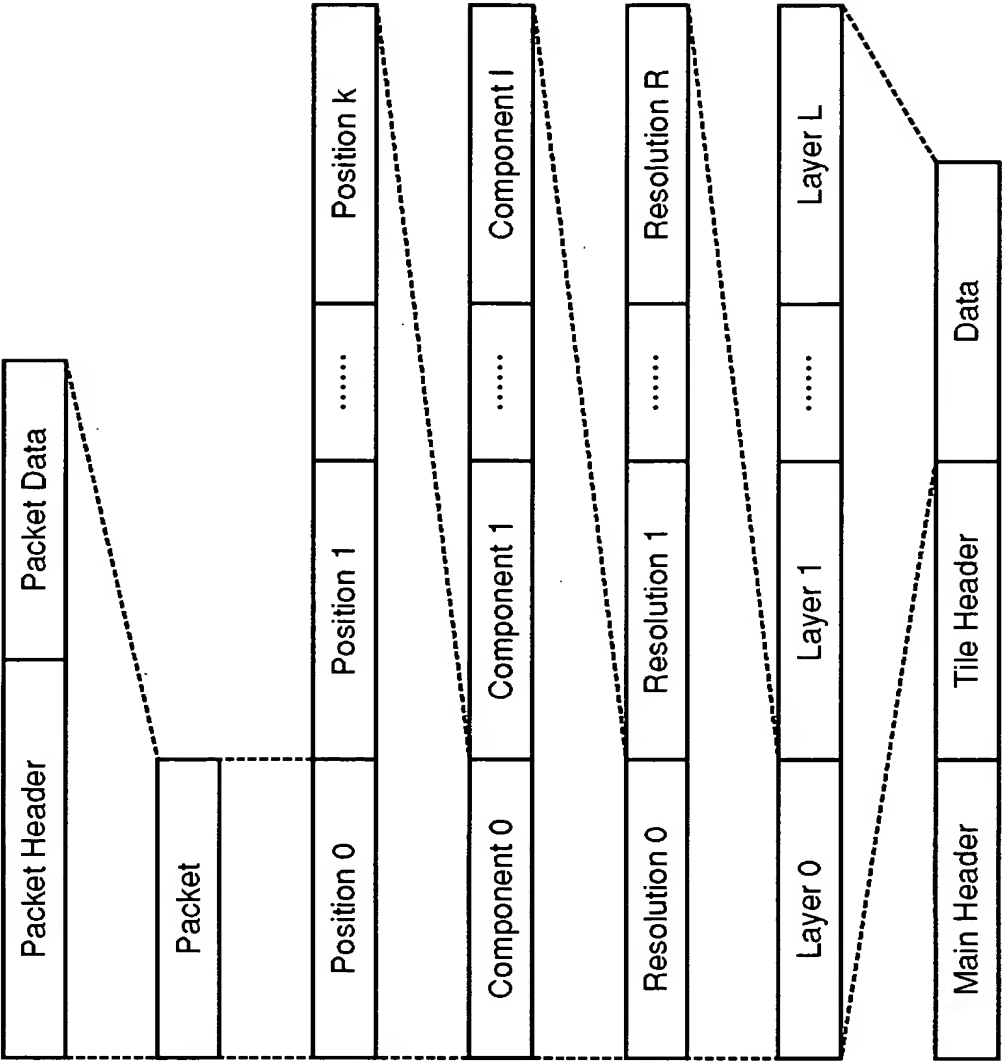
【図 1】



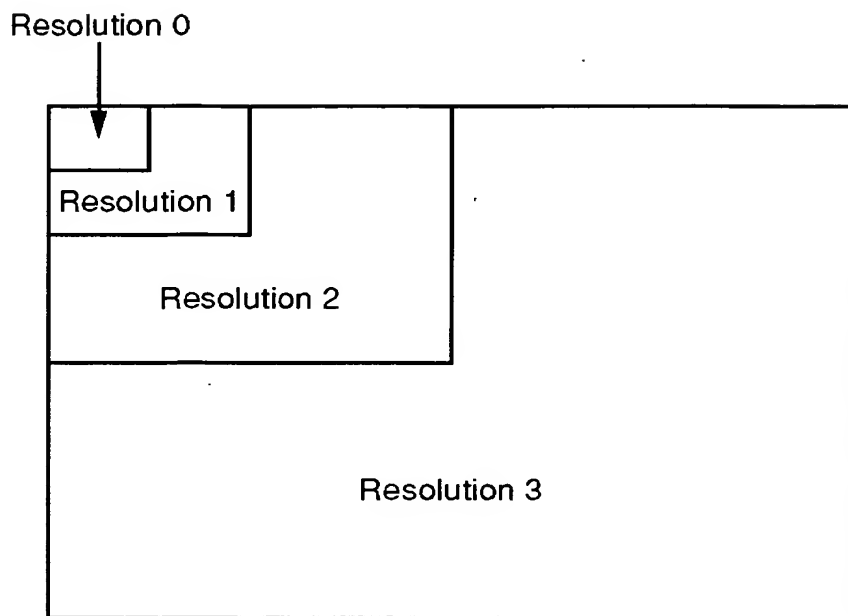
【図 2】



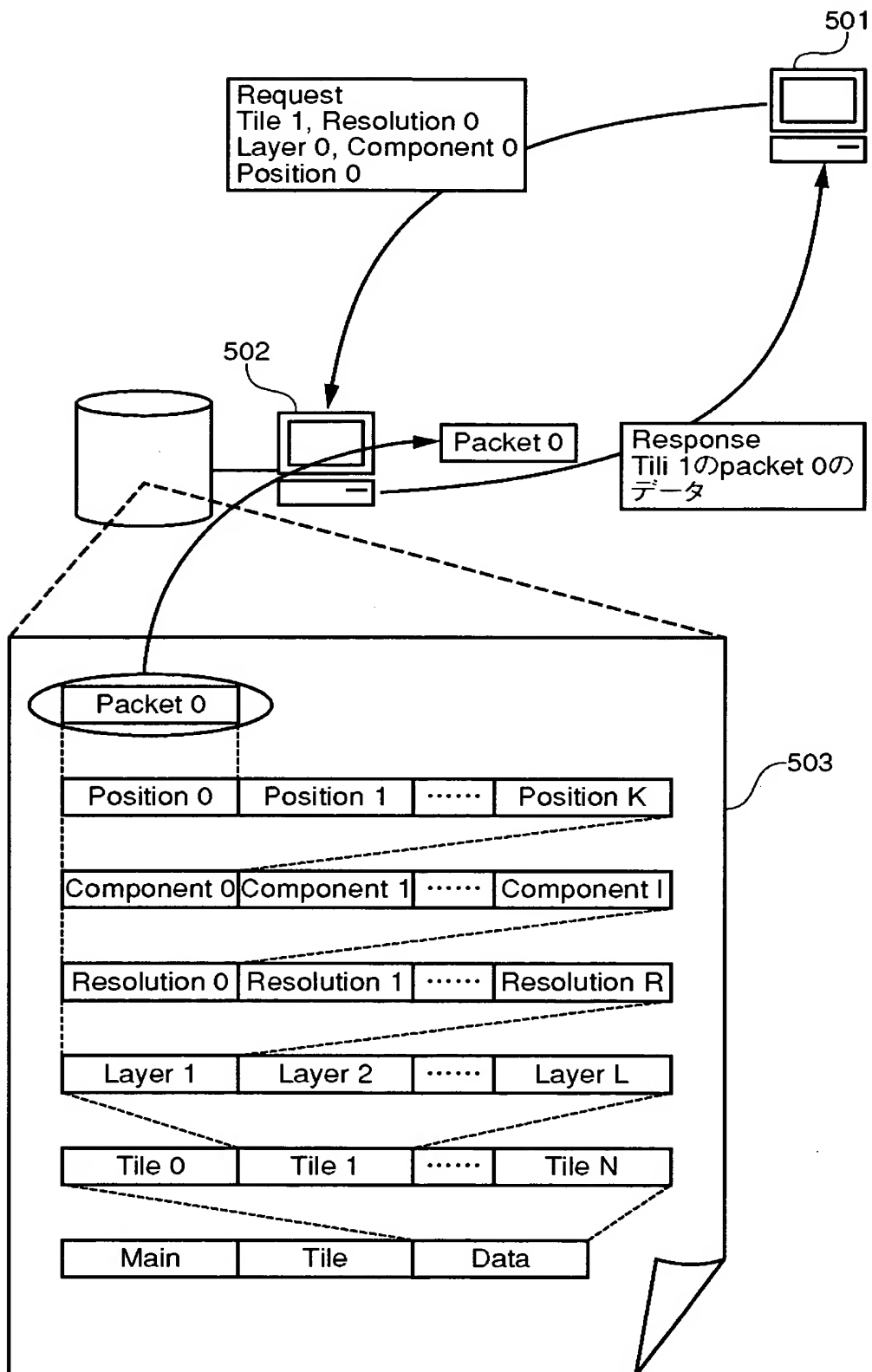
【図 3】



【図 4】



【図 5】



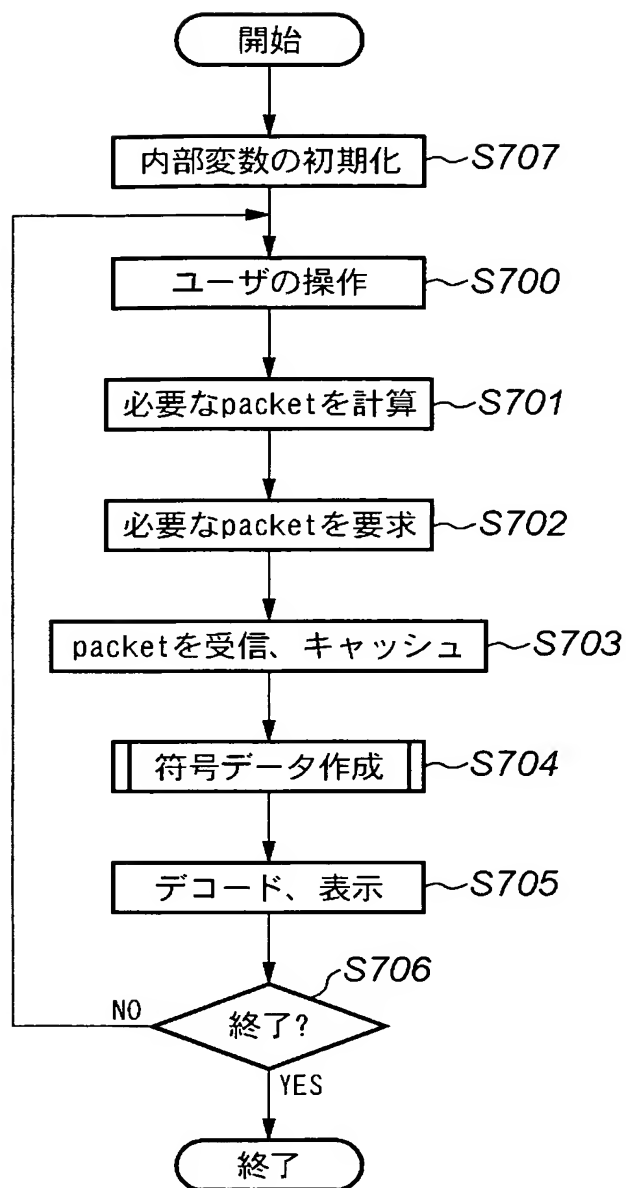
【図 6 A】

Tile 0	Tile 1	Tile 2	Tile 3
Tile 4	Tile 5	Tile 6	Tile 7
Tile 8	Tile 9	Tile 10	Tile 11
Tile 12	Tile 13	Tile 14	Tile 15

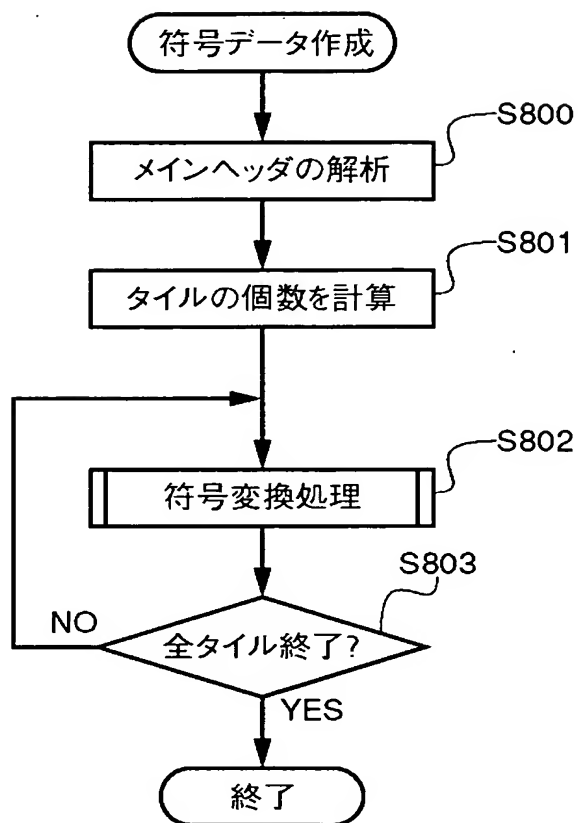
【図 6 B】

Main Header	SIZマーカ・コード	Xsize = 1024
	CODマーカ・コード	Ysize = 1024
	QCDマーカ・コード	XTsize = 256
Tile part Header 0	SOTマーカ・コード	YTsize = 256
	SODマーカ・コード	
符号化データ	Lay0/Res0/Com0/Pos0	
	Lay0/Res1/Com0/Pos0	
	Lay0/Res2/Com0/Pos0	
	Lay0/Res3/Com0/Pos0	
	Lay1/Res0/Com0/Pos0	
	Lay1/Res1/Com0/Pos0	
	Lay1/Res2/Com0/Pos0	
	Lay1/Res3/Com0/Pos0	
	Lay2/Res0/Com0/Pos0	
	Lay2/Res1/Com0/Pos0	
	Lay2/Res2/Com0/Pos0	
	Lay2/Res3/Com0/Pos0	
Tile part Header 1	SOTマーカ・コード	
	SODマーカ・コード	
符号化データ		
:		
Tile part Header 15	SOTマーカ・コード	
	SODマーカ・コード	
符号化データ		

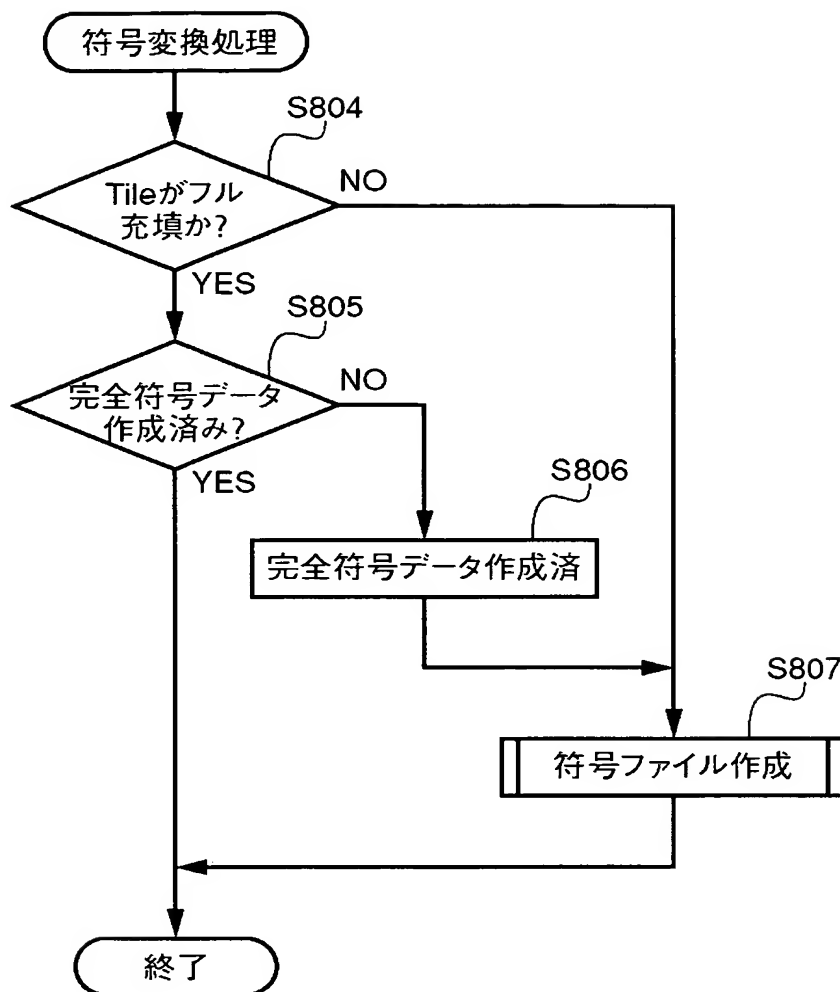
【図 7】



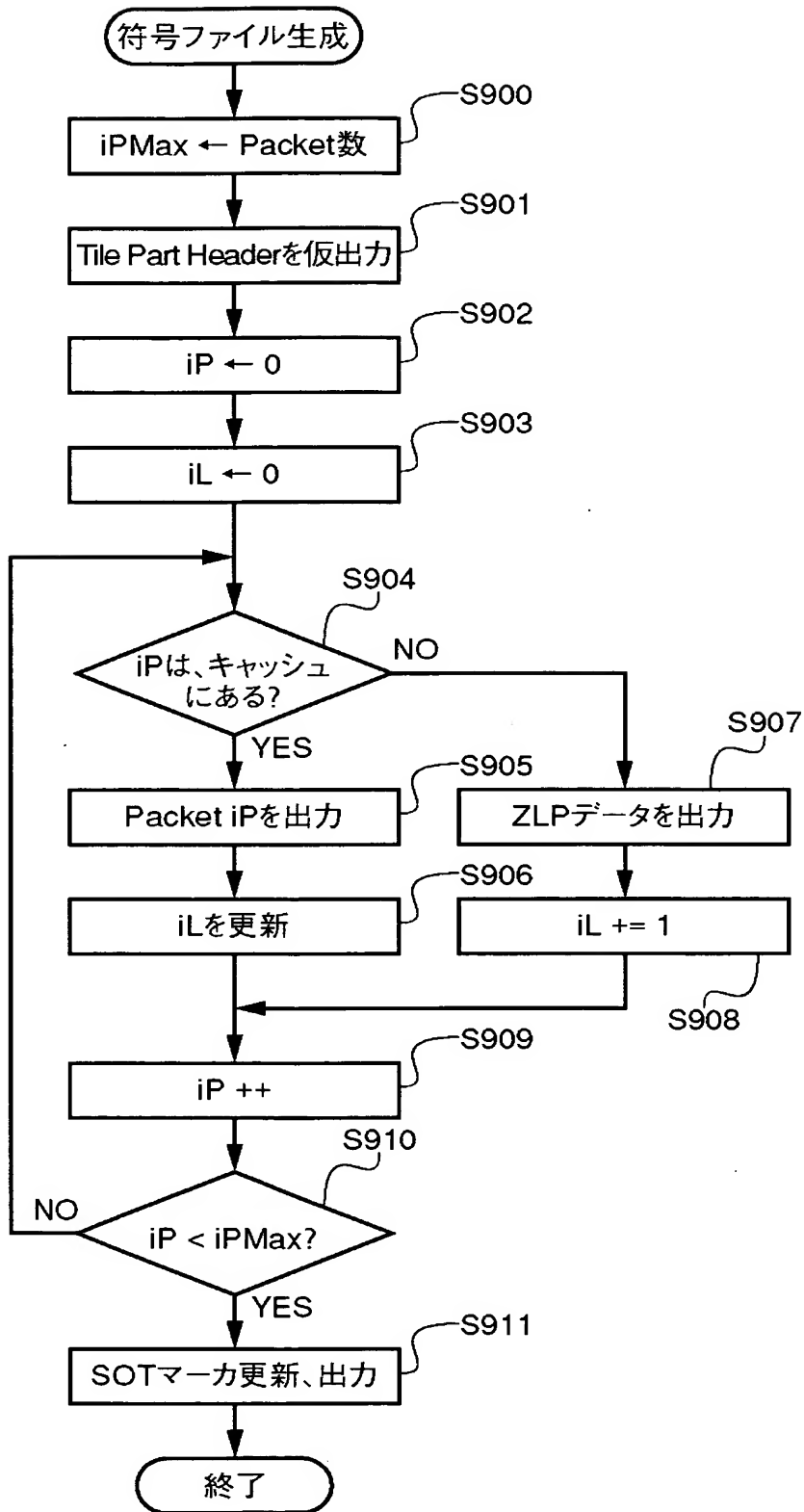
【図 8 A】



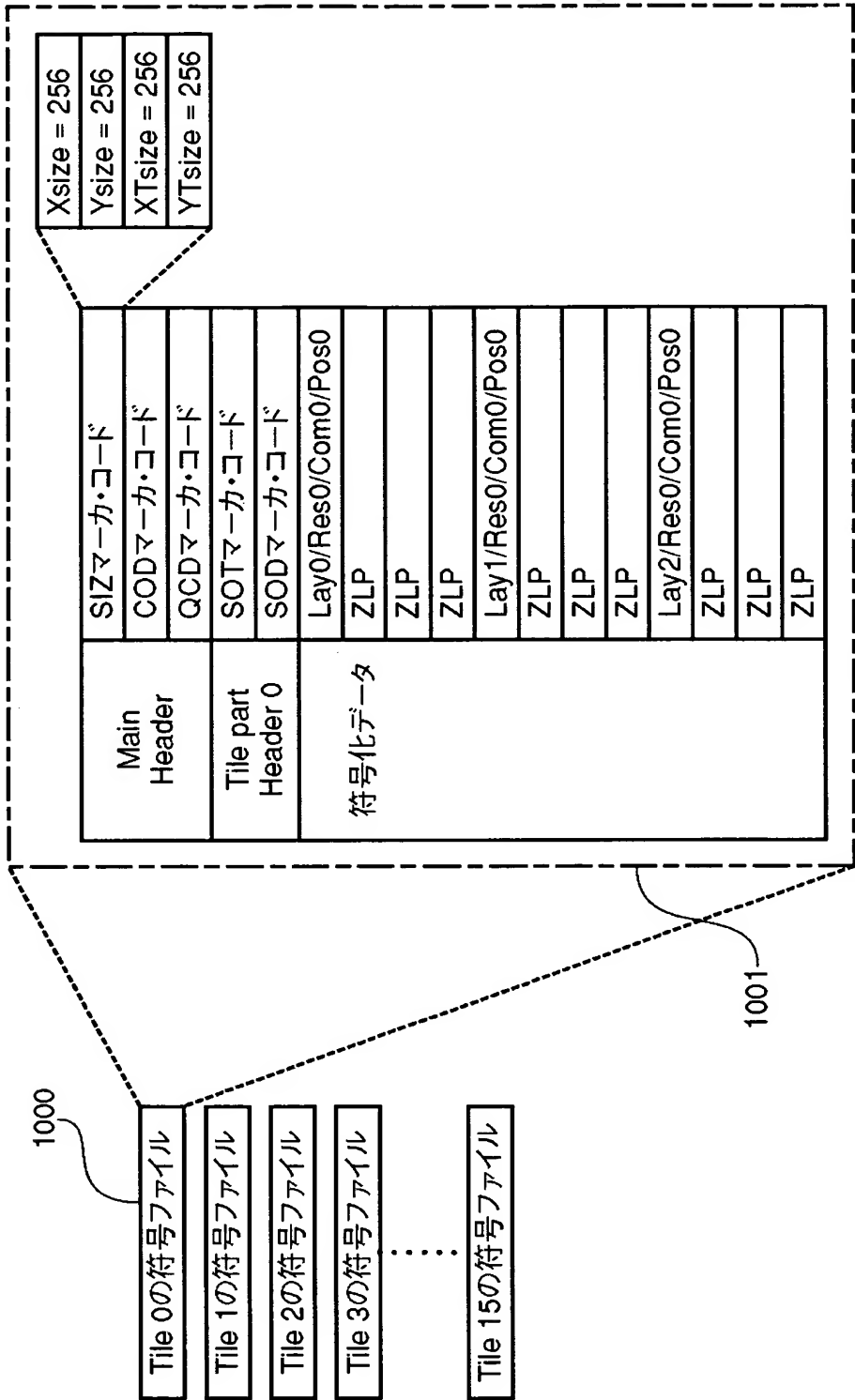
【図 8 B】



【図 9】



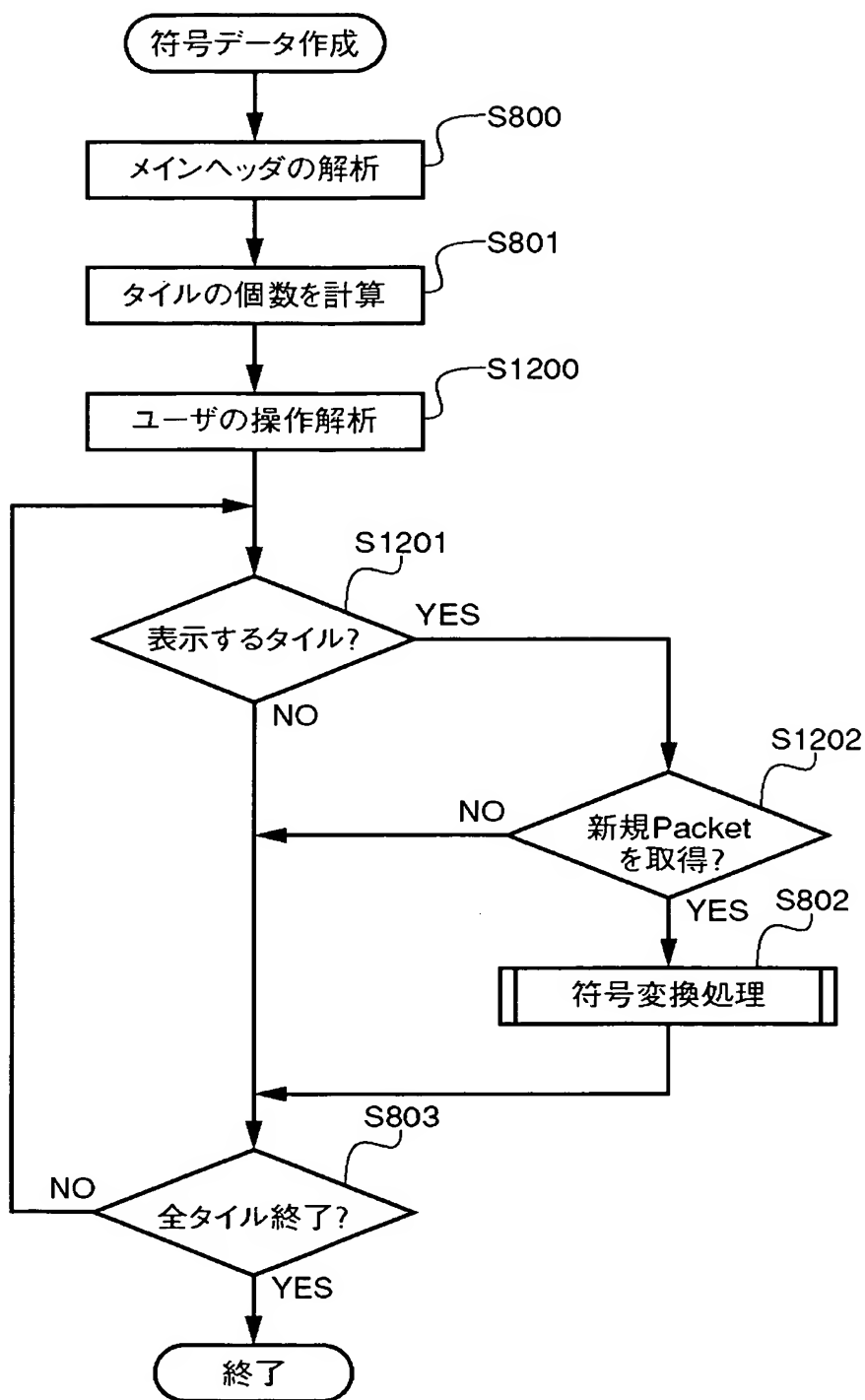
【図 10】



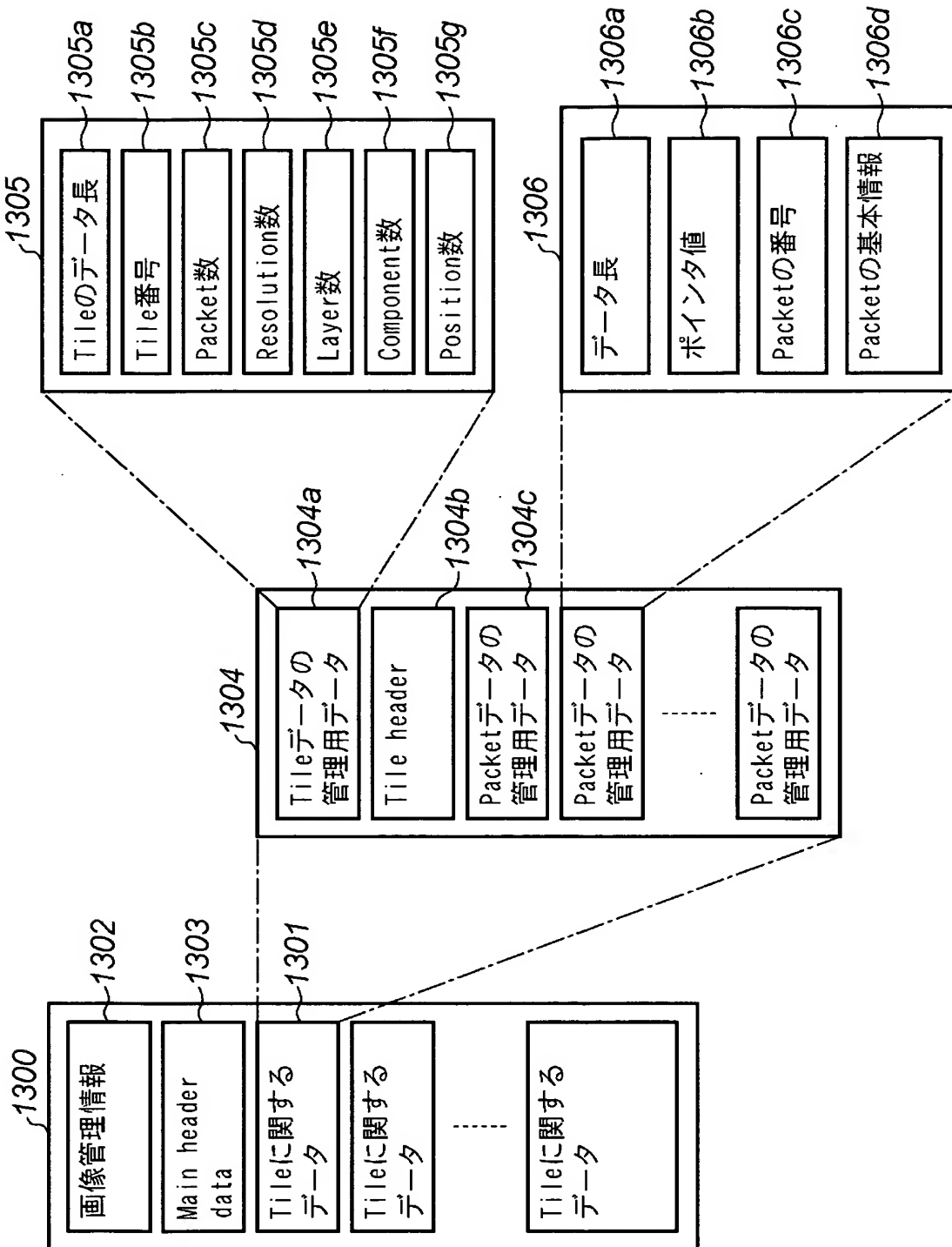
【図 11】

1001		1101		1102	
Main Header	SIZマーカ・コード	Main Header	SIZマーカ・コード	Main Header	SIZマーカ・コード
	CODマーカ・コード		CODマーカ・コード		CODマーカ・コード
	QCDマーカ・コード		QCDマーカ・コード		QCDマーカ・コード
	SOTマーカ・コード		SOTマーカ・コード		SOTマーカ・コード
	SODマーカ・コード		SODマーカ・コード		SODマーカ・コード
Tile part Header 0	Lay0/Res0/Com0/Pos0	Tile part Header 0	Lay0/Res0/Com0/Pos0	Tile part Header 0	Lay0/Res0/Com0/Pos0
	ZLP		Lay0/Res1/Com0/Pos0		Lay0/Res1/Com0/Pos0
	ZLP		Lay0/Res2/Com0/Pos0		Lay0/Res2/Com0/Pos0
	ZLP		ZLP		Lay0/Res3/Com0/Pos0
	Lay1/Res0/Com0/Pos0		Lay1/Res0/Com0/Pos0		Lay1/Res0/Com0/Pos0
	ZLP		Lay1/Res1/Com0/Pos0		Lay1/Res1/Com0/Pos0
	ZLP		Lay1/Res2/Com0/Pos0		Lay1/Res2/Com0/Pos0
	ZLP		ZLP		Lay1/Res3/Com0/Pos0
	Lay2/Res0/Com0/Pos0		Lay2/Res0/Com0/Pos0		Lay2/Res0/Com0/Pos0
	ZLP		Lay2/Res1/Com0/Pos0		Lay2/Res1/Com0/Pos0
	ZLP		Lay2/Res2/Com0/Pos0		Lay2/Res2/Com0/Pos0
	ZLP		ZLP		Lay2/Res3/Com0/Pos0
符号化データ		符号化データ		符号化データ	

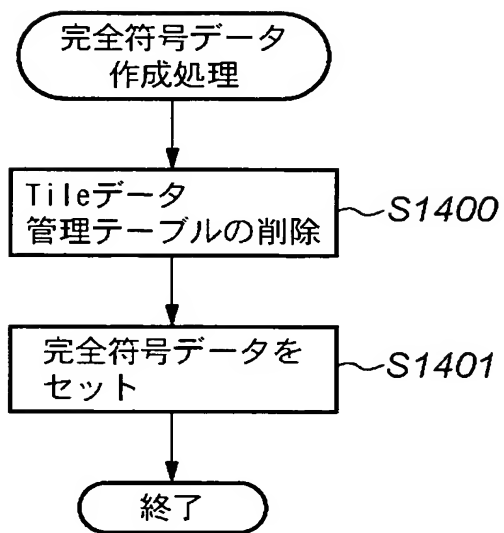
【図 12】



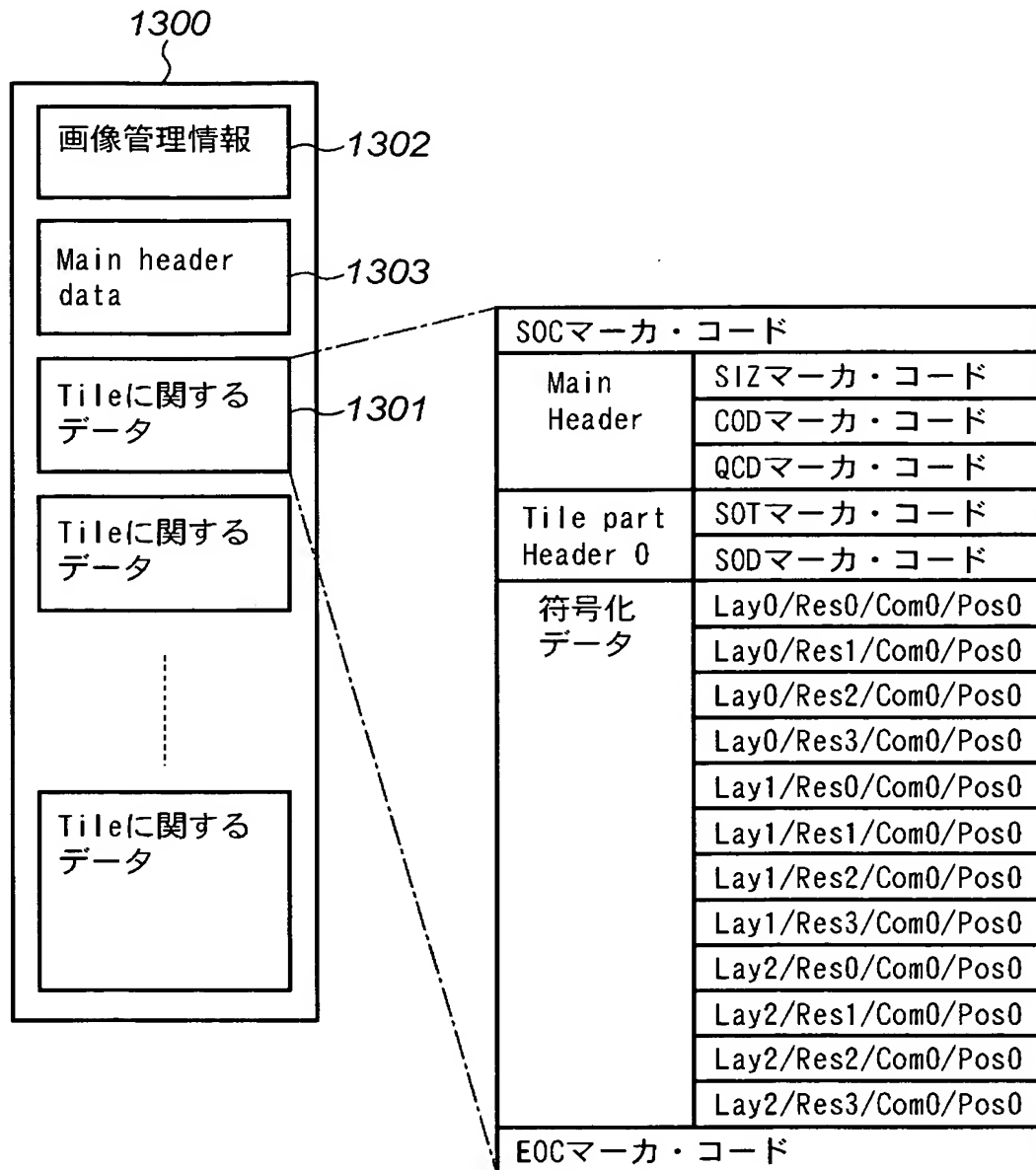
【図 13】



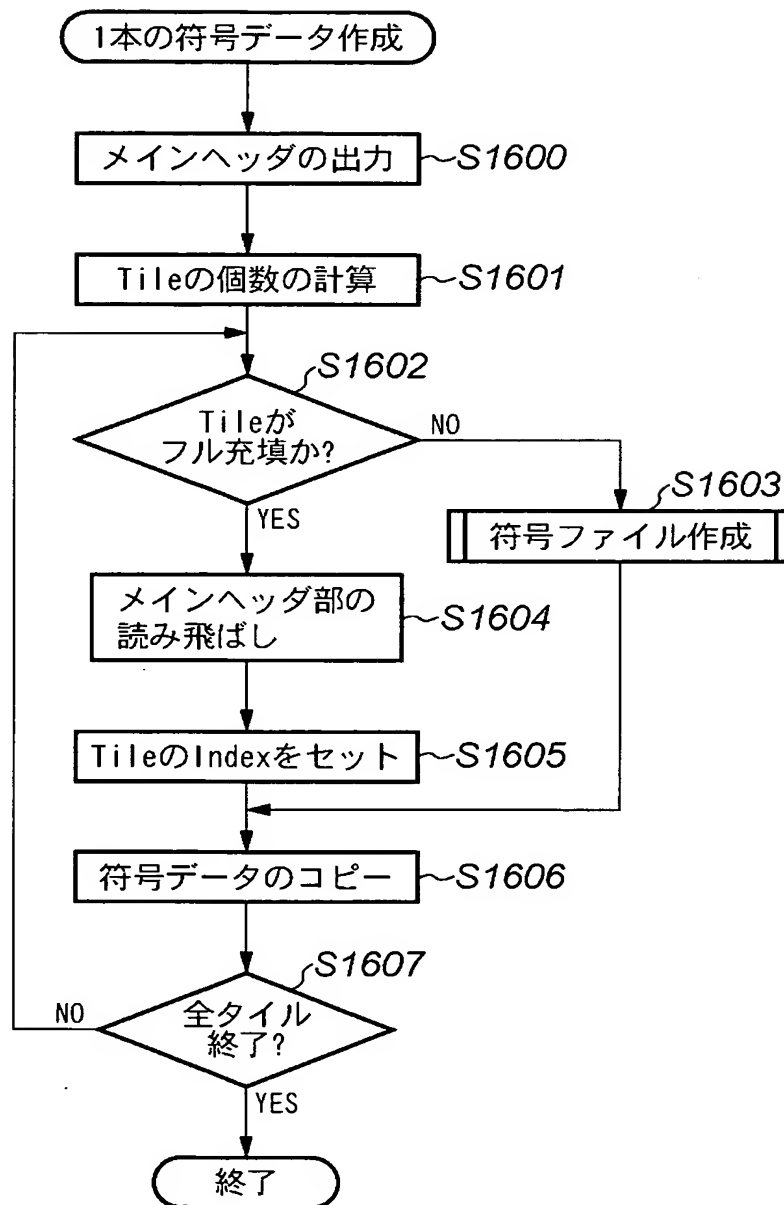
【図 14】



【図 15】



【図 16】



【書類名】 要約書

【要約】

【課題】 クライアントにキャッシュされている断片化された符号データと、サーバから必要な分だけ受信した断片化された符号データとを用いて、クライアントにおいて汎用 J P E G 2 0 0 0 デコーダで使用可能な符号データを好適に作成することができる符号データ作成方法及び装置を提供する。

【解決手段】 クライアントは、サーバが管理する符号データのうち、第 1 の符号データを格納しており、J P E G 2 0 0 0 符号データの作成に必要となる符号データと第 1 の符号データとから、不足する第 2 の符号データを算出する。そして、サーバから第 2 の符号データを取得し、そのヘッダ情報を解析して、符号データを複数の独立した符号データに分割する。さらに、分割された単位毎に、独立した符号データの全てのデータが格納されていない場合、ダミー符号データを格納し、その符号データを J P E G 2 0 0 0 符号データとする。

【選択図】 図 5

認定・付加情報

特許出願の番号	特願 2003-201162
受付番号	50301223647
書類名	特許願
担当官	小野寺 光子 1721
作成日	平成 15 年 7 月 29 日

< 認定情報・付加情報 >

【特許出願人】

【識別番号】	000001007
【住所又は居所】	東京都大田区下丸子 3 丁目 30 番 2 号
【氏名又は名称】	キャノン株式会社

【代理人】

申請人

【識別番号】	100076428
【住所又は居所】	東京都千代田区紀尾井町 3 番 6 号 秀和紀尾井町 パークビル 7 F 大塚国際特許事務所
【氏名又は名称】	大塚 康德

【選任した代理人】

【識別番号】	100112508
【住所又は居所】	東京都千代田区紀尾井町 3 番 6 号 秀和紀尾井町 パークビル 7 F 大塚国際特許事務所
【氏名又は名称】	高柳 司郎

【選任した代理人】

【識別番号】	100115071
【住所又は居所】	東京都千代田区紀尾井町 3 番 6 号 秀和紀尾井町 パークビル 7 F 大塚国際特許事務所
【氏名又は名称】	大塚 康弘

【選任した代理人】

【識別番号】	100116894
【住所又は居所】	東京都千代田区紀尾井町 3 番 6 号 秀和紀尾井町 パークビル 7 F 大塚国際特許事務所
【氏名又は名称】	木村 秀二

特願 2 0 0 3 - 2 0 1 1 6 2

出 願 人 履 歴 情 報

識別番号

[0 0 0 0 0 1 0 0 7]

1. 変更年月日

1 9 9 0 年 8 月 3 0 日

[変更理由]

新規登録

住 所

東京都大田区下丸子 3 丁目 3 0 番 2 号

氏 名

キヤノン株式会社